第2章

新人研修は「模型ロケット」《新人技術者編》

――ソフトウェア会社の新入社員5名，嵐の中の旅立ち

渡邉友裕，杉山 歩，沼田亜美，後藤孝一，京井真之



ここではソフトウェア開発会社（ヴィッツ）の新人研修の事例を紹介する．新人技術者の５人には「模型ロケットを飛ばして，搭載したセンサで加速度や地磁気を観測・記録する」という課題が与えられた．５人は，ロケット担当，基板担当，デバイス・ドライバ担当，OS 担当，アプリケーション・ソフトウェア担当に分かれて作業を進めた．はたして，模型ロケットの打ち上げは無事成功したのだろうか…．（編集部）

2004年に開催されたSWEST6（Summer Workshop on Embedded System Technologies；組込みシステム技術に関するサマーワークショップ）において，小型の模型ロケット（いわゆるモデル・ロケット）を利用した低高度からの地上観測実験「Hamana-1」が実施されました．そしてその続編として，2005年8月末開催のSWEST7では，観測機器を搭載した模型ロケットの発射コンテスト「Hamana-2」が企画されました．打ち上げ場所は浜名湖（静岡県浜松市）です．



写真1
模型ロケットの専門書
「飛ばせ！手作りロケット－Flying Model Rocket」，日本モデルロケット協会 編集，誠文堂新光社 刊（ISBN 4-416-39905-7）．最終的にこの書籍を丸暗記するほど読むことになる．内容の不明点について，モデルロケット協会と何度かやりとりすることもあった．

某部署に配属になった私たち新入社員は，新人研修の一環として，「Hamana-2」に参加することになっていました．が，入社したばかりの私たちは当時，そんなことを知る由もなく….

1 Hamana-2 ロケット・プロジェクトの軌跡

2005年5月6日，5名の新人が某部署に配属されることとなり，先輩エンジニアからすべての発端となることばが浴びせかけられました．「お前たち新人だけで，観測機器を搭載した模型ロケットの打ち上げを成功させろ」．

そのときに提示された要求は，「ロケットに機器を搭載して打ち上げ．打ち上げ中の加速度/地磁気を観測し，そのデータを記録．そして，無事ロケットを回収し，記録したデータを解析．打ち上げの軌跡なども表示するようなアプリケーションも作れたらいいな」というものでした．そのことが，いかに困難なことなのか．前途にどのようなことが待ち受けているのか．まだ人生のぬるま湯につかりきっていた私たちには想像もできなかったのです．

● あまり調査しないまま，試射に成功

とりあえず模型ロケットがどのようなものなのかがわかっていなかったので，「観測機器をどうするか」を検討する以前に，模型ロケットの打ち上げを行うところから始めました．最初に小型の模型ロケットの作成キットと，「打ち上げかたとか，詳しいことはこれに書いてあるから」と言

KeyWord　モデル・ロケット，Hamana-2，SWEST，デバイス・ドライバ，TOPPERS，H8，EEPROM，SPI，Pizzafactory，Cygwin

われながら，ロケットの専門書を渡されました（**写真1**）．「会社の新人研修でこんなことをやっていていいのかな？」と心の中でつぶやきながらロケットを作成していたことを覚えています．

ロケットの専門書は全部で100ページくらいの厚さなのですが，この本の内容を暗記するくらいまで熟読することになるとは，このときは思ってもいませんでした．不慣れな手つきでなんとか模型ロケットを作成し，いざ打ち上げることになりました．

初めての試射の段取りは，すべて先輩エンジニアの方々に行ってもらったのですが，今思えば，これが大失敗への布石になってしまったと感じています．先輩方の段取りが良かったため，私たちは打ち上げについてほとんど調べないまま，初めての打ち上げに成功してしまったのです（**写真2**）．

● 漠然とデータシートとにらめっこ

ロケットを無事打ち上げることに成功し，次は「観測機器をなんとかしよう！」ということになったのですが，漠然としたイメージだけで，要求を満たすために何をするべきか，何が必要なのかわかるわけもありません．

先輩エンジニアに言われるままに，今回使用するセンサやマイコンの調査ということで，データシートとにらめっこする日々が続きました．データシートを読んでもどこが重要なのかまったく検討がつかず，どこがポイントなのかわからないまま読んでいるため，読めども読めども理解度は上がらず，時間だけが過ぎていきます．わからない点について先輩に質問しに行けばよかったのかもしれませんが，しごとをしている先輩たちとは別に，新人だけが別室に集められ，日ごろあまり顔を合わせていませんでした．そのため面識があまりなく，声がかけづらかったのです．

先輩や上司の管理の目がないことからモチベーションも上がらず，作業の進捗は遅れていく一方でした．また，遅れることに対する危機感もありませんでした．なんとかなるだろうと思い，疑いませんでしたし，困ったら先輩が何とかしてくれるだろうと思っていたのです．

一応，週2回の「意識合わせ＋進捗報告」のミーティングがあったのですが，先輩エンジニアと新人の中の各担当者（ロケット担当，基板担当，デバイス・ドライバ担当，OS担当，アプリケーション・ソフトウェア担当に分かれていた）がわからないことを1対1で話し合い，その間，ほかの新人はポカンとしている状況でした．先輩エンジニアはみんなの情報を知ってはいるものの，新人どうしの知識の共有というものはなく，一つのプロジェクトにかかわっている者どうしなのに個別に動いていました．それでも，「まぁ，みんな何かやってるみたいだし，いいか」くらいの考えでいました．

そのような状態で第2回目の試射を行うことになりました．今回の課題は，「新人のメンバだけで打ち上げの準備を行う．前回は低出力のロケットだったが，今回からは本番で使用する，高出力でより高く飛ぶロケットを使う．観測機器を搭載することを想定し，代わりに重りをロケットに積んで打ち上げる」というものでした．

前回，事前の準備がほぼゼロで打ち上げが成功したため，今回もほとんど事前の準備を行わず当日を迎えました．



◀ **写真2**
模型ロケット初回打ち上げの風景
模型ロケット（ALPHA3）の初打ち上げに成功した．



▶ **写真3**
準備不足で迎えた2回目の打ち上げ風景
前回の打ち上げよりも機体が大きく，エンジンも高出力にした模型ロケット（BABY BERTHA）を打ち上げた．バランスの調整も，リカバリ・システムのセッティングもめちゃくちゃだった．この後，ロケットは吸い込まれるようにやぶの中へと消えていった．



写真4　やぶの中へと姿を消した模型ロケットの捜索
炎天下の中，必死に身の丈ほどもある草をかき分けて，20分以上も捜索を続けた．しかし，2度とその姿を見ることはなかった．

● 悪夢の限界領域

打ち上げ日当日，ついにボロがでました．各担当者の情報共有ができていないことがここで露呈しました．ロケット打ち上げの際に必要なセッティングについて，ロケットの担当者しかわかっていなかったのです．そのため，ロケット担当が当日のすべての作業を行うことになりました．それ以外のメンバは何をしてよいかわからないので，右往左往するだけです．また担当者も，事前の準備ができていなかったため，打ち上げ場所に到着してから，打ち上げに必要な部品を忘れたことに気づき，ようすを見に来た先輩エンジニアにすがって部品を借りる始末でした．

なんとか，打ち上げられる状態までもってはいけたものの，準備の段階でかなりの時間がかかってしまいました．ロケットが風で流されないように，風のない早朝をねらって全員が集合したのですが，準備が終わったのは昼近く．打ち上げるころには，徐々に風がでてきてしまいました．それでも何とか威力の強いC型エンジンによる打ち上げを成功させ，いよいよロケットに重りを載せて打ち上げることになりました（**写真3**）．そのころには風はかなり強くなっていました．そして惨劇が起きました．カウントダウン開始，5，4，3，2，1…イグニッション（点火）．

「あーーーーー！！」．

みんなの悲鳴が聞こえました．ロケットは真っ直ぐではなく斜めに打ち上がり，そのままやぶの中へと消えていきました．ロケットをロスト（消失）してしまったのです（**写真4**）．ロケットはバランスが重要であるにもかかわらず，重りを積んだときの機体のバランスについて何も調べていなかったので，当然の結果といえばそうなのですが…．また，本来ならば風が強い場合はロストする確率が非常に高いため，打ち上げそのものを中止するべきでした．

ほんとうに，絵に描いたような失敗とはまさにこのことだと思いました．「情報共有不足，事前調査不足，考察不足」．成功するわけがありません．

大失敗を犯してしまい，上司から本プロジェクトそのものを続けるかどうか，私たちだけで考えるように言われました．くしくもプロジェクトが始まってからちょうど1ヵ月．新人どうしで真剣に話し合うのは，実はこれが初めてでした．そのときからです．みんなが必死になって，本気でプロジェクトを成功させようと考えるようになったのは…．

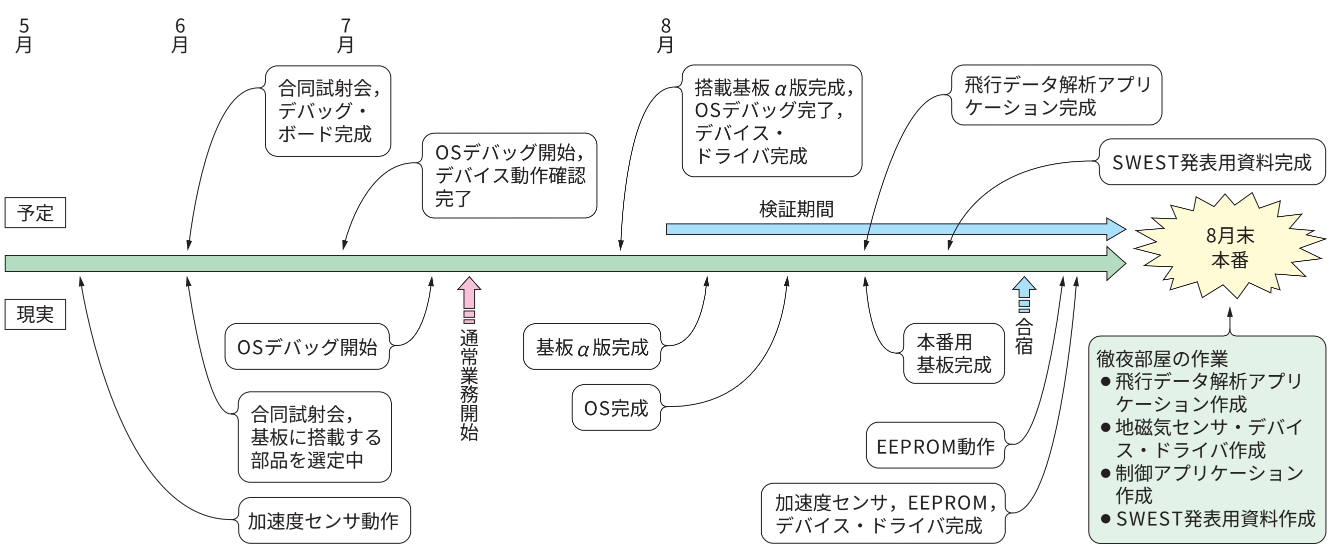


図1　ロケット・プロジェクトの予定と実際の進捗
当初の予定では8月にはすべてのシステムが完成し，検証フェイズに入っている予定だった．

● 進まないロケット・プロジェクト

最初の試射を終え，メンバ全員が気合いを入れ直してプロジェクトに取り組んだのですが，みんな，初めての作業です．調査にばかり多くの時間が取られ，思うように進みませんでした．6月中旬にはHamana-2参加チーム合同の試射会が行われました．ここで完成したデバッグ・ボードとシステムの構想について，ほかのチームに報告する予定だったのですが，実際にはボードは完成せず，たいした報告もできないまま終わってしまいました．

予定では8月の初めにはひと通りの作業が完了して，調整・検証作業を行っているスケジュールだったのですが，そううまくいくはずもありません．各担当者がやっとそれらしく作業を行えるようになったのが，7月の終わりでした(**図1**)．

● 舞い降りた通常業務

しかし7月後半になると，会社の通常業務が非常に忙しくなってきました．ロケット・プロジェクトと通常業務のかけ持ちで作業を進めなければなりません．私たちもやっとそれなりに作業ができるようになり，「さぁ，ここからだ！」という気分だったのですが，当然のことながら通常業務のほうが優先であったため，ロケット・プロジェクトの進捗はこれまで以上に遅れていきました．このころの作業報告を見返すと，全員がロケット・プロジェクトにほとんど時間を割けていませんでした．

作業が進まないままどんどん時間は過ぎ，SWEST7本番まであと10日．動作が不安定な基板，正常に動作してくれないOS，沈黙を続けるデバイス・ドライバ….ロケットは無事に空を飛ぶのでしょうか？

● 時間との戦い，自分との戦い

このプロジェクトが立ち上がった当初の予定では本番の1週間前，8月19日の早朝に最終試射を行う予定でした．しかし，試射予定日の前々日になっても計測用のシステムがまったくと言っていいほど完成していませんでした．これはどう考えても間に合わないだろうということで話し合った結果，合宿が決定！！ 新人5人プラス指導していた先輩エンジニアが徹夜で作業することになりました．

システムを作る担当のメンバは焦りに焦ります．先輩からいろいろと指導してはもらうものの，焦るほど作業ははかどらず，時間が過ぎていきます．午前3時，4時，5時，そして小鳥が鳴きだし，日の光が見えたとき，「ついに！ついに！待望のシステムが完成！！」……と言いたいところなのですが，物語のようにそううまくいくことはなく，システムは未完成のままでした．現実の厳しさを目の当たりにしながら，さすがに疲れはててみんな会社で睡眠をとりました．

ほんの少し寝て，また作業を開始．しかし，寝不足で頭が働かない．さらに焦りだけが先走りして作業が進まず，次の夜を迎えるがいまだ完成する気配なし．最終試射はあすなのに，肝心のシステムが完成しません．そのまま朝まで作業が続くのですが，前日の睡眠不足から体力・気力の限界に達し，もう作業を続けるのは困難な状況です．未完成でこのまま作業を続けても効率が悪いと判断し，予定日の最終試射をあきらめることになりました．

● 限界ギリギリ

一度しっかり寝てから，しきり直しました．無理な徹夜はせず，システムを作り上げることに専念する方針に転換しました．そして，なんとか加速度だけ計測できるものができたのが，SWEST7本番の前々日です．当初予定していたシステムと比べるとクオリティはかなり低いものにはなりましたが，Hamana-2企画の要件の一つである「観測機器を搭載した模型ロケット」というのはなんとか満たしています．

ほっとひと息つきたいところですが，そんな暇はありません．あす1日しかない試射のために大急ぎで準備を始めました．

本番前日の早朝，観測機器を積み込んで打ち上げる最初で最後の試射．天気は晴れ．風ややあり．絶好と言えるコンディションではないものの，雨が降らずひと安心．予定では4発の打ち上げです．本番は1発ですが，計測に失敗したら意味がないので，余裕を持って4発試射することにしました．この4発の試射が，後でとんでもない事態を引き起こすことになるのです．

ロケット担当者が機器を慎重にロケットに搭載していきます．まわりはみんなドキドキ．打ち上げについては，今までの失敗の経験を生かして準備は万全．そしてついに1発目の打ち上げです．ロケットは真上に青空へ向かって打ち上がりました．打ち上げに関しては文句なしの成功です．ただし，きちんと計測できたかどうかが不安でした．その場で確かめる機材は持ってこなかったので，記憶装置だけを取り替えて2発目の試射の準備にとりかかりました．

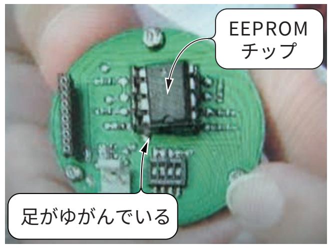


写真5
EEPROMチップの足がゆがむ
ロケット墜落の衝撃により，EEPROMの足の部分がゆがんだ．



写真6　徹夜部屋の風景
デバッグのためにオシロスコープや安定化電源まで持ち込み，頭をかかえてテンパっている．

2発目，3発目もそれまでと比べものにならないくらいまっすぐに，真上に打ち上がりました．そして運命の4発目を打ち上げるときがきました．少々斜めに上がったけれど許容範囲内の角度で打ち上がり，「成功！」と思った瞬間，高速落下を始めるロケット．ん？ パラシュートが開いてないではありませんか！！

「あー！！」

叫んでいる間に地面にたたきつけられるロケット．急いで駆け寄る私たち．無残にもロケットの羽根はもぎ取られ，搭載した基板はゆがみ，壊れていました（写真5）．

実際には4回の打ち上げで2回分のデータが取れたのですが，あとでソース・コードを見直したところ，致命的なミスが発覚しました．さらに基板が壊れるという最悪の事態です．完成を目指して焦りに焦って修正作業を行いました．そして未完成のまま，SWEST7当日に…．

● SWEST7で最後の試練

さてSWEST7本番当日，ワクテカ気分でSWESTを迎えているはずの予定が，メンバ一同，焦りと疲れで肉体的にも精神的にもボロボロです．いろいろな発表やセッションが行われる中，頭の中はシステムのことでいっぱいでした．時間を見つけてはノート・パソコンで作業するという状況です．そして夜，再び悪夢が訪れることになります．

懇親会も終わり，“徹夜部屋”と呼ばれる地獄（？）の部屋で新人一同は修正作業を進めます（写真6）．本番前日に基板が壊れてしまったため，予備で作っておいた基板でシステムをデバッグしていたのですが，なぜか昨日のように記憶装置に値が書き込めないのです．なぜなのかよくわからずソース・コードを眺めていると，先輩エンジニアがひと言，「このソース，古くないか？」．デバイス・ドライバ担当の新人がたしかに古いことを確認しました．そういえばさっきファイル操作したな…．

なんと，最新のソースに古いソースを上書きしてしまったらしいのです．そのうえ，上書きする前にバックアップをとらなかったようで，ファイルが見つかりません．もう笑うしかありません．このとき夜の0時．打ち上げまであと9時間．こうなってはどうしようもないので，またソースを作り直すことになりました．

時間が過ぎていく中，なかなかソースを元のように戻すことができず，朝を迎えました．もうダメかとみんなが思ったとき，メンバのひとりがふと思い出しました．「システムのソースって，会社のWebサーバに上げてあるのでは？」．

「これだ！」ということで，早速ダウンロードを試みました．「やったー，これでなんとかなるぞ」と思いましたが，今度はここにきて基板が不調になりました．もともと動作が不安定だった自作基板が，ついに限界に達したようです．基板の復旧を試みましたが，さすがにもう時間がなく，ドタバタ劇もここで幕を閉じることになりました．

しかし，最後の最後まで騒動は続きます．打ち上げの前日に台風が日本列島を直撃し，浜名湖で打ち上げたロケットを回収するボートを出すことができないということで，打ち上げそのものが中止になったのです．さすがに浜名湖まで来て何もせずに帰るのはなんだということで，有志だけで打ち上げを決行することになりました．しかし，システムが完成していないのでロケットだけの打ち上げに…．なんとも悔いの残る結果となって，私たちの新人研修は幕を閉じました．

2 各担当者の作業の詳細

私たち5人は，それぞれロケット担当，基板担当，デバイス・ドライバ担当，OS担当，アプリケーション・ソフトウェア担当に分かれて作業を行いました．以下では，各担当者が実際に行った作業内容について記述していきます．

● ロケット担当：試射場所の確保で役所にかけ合う

ロケット担当（渡邉）にとっては，模型ロケットに詳しい先輩エンジニアが会社にいなかったため，知りたい情報に詳しい人物が近くにいないときにいかにして情報をかき集めるかということが問題となりました．

まず一つ目の課題は，本番の前の試射の場所を探すことでした．模型ロケットは上空100m～200mまで上昇するため，安全面を考えてかなり広い平地でないと打ち上げを行うことができません．なかなかそのような場所がなかったのですが，国土交通省と交渉の末，なんとか試射場所を確保しました．

また，今回はロケットを打ち上げたあと，上空でパラシュートを開いて回収を試みたのですが，当初はうまく展開しませんでした．本を調べても最低限のことしか載っておらず，途方にくれることになりました．最終的にはモデルロケット協会に連絡をとり，パラシュートのセッティング方法について，いろいろと教授していただきました．

また，基板担当者とよく議論しました．これは，搭載する基板の重さがロケットのバランスに影響を与えるためです．この重さのときはだいじょうぶか，この場所に取りつけて打ち上げ時の安定性に問題はないかなど，何回も情報交換を行いました．担当は別でも，いろいろなことが密接にかかわってくることが，このプロジェクトを通してわかりました（写真7）．

ロケット担当としては，開発というより，ひたすら調査ばかりしていた気がします．

● 基板担当：試行錯誤を繰り返し，完成した基板は2組

「ロケットに搭載する回路は全部自分で作って」と先輩エンジニアにさらっと言われたところから，基板担当者（杉山）の作業はスタートしました．しかし，担当者はプリント基板の設計などしたことがありません．まずは，基板を製作するためにはどのような工程が必要かというところから調べていきました．

1）まずは回路設計

動作する回路を早く作らないと，ソフトウェア担当の人たちが困ってしまいます．そのため，とりあえずデバイス・ドライバ担当にデバッグ用ボードを渡すことを目標に，回路設計を始めました．が，案の定なかなかうまくいきません．マイコンを単体で動作させる回路や，各デバイスとマイコンとの通信回路など，わからないことだらけでした．その中でもいちばん困ったのが，マイコン（5V動作）と地磁気センサ（3V動作）の間の信号レベルの変換です．試作回路がうまく動作せず，動作を確認しては回路を作り直す，という作業を繰り返すことになりました．

2）どうやってロケットに載せるか

回路設計が落ち着いてきたころ，本番で使用する模型ロケットが私たちの前に届きました．ひと目見た感想は，「小さい！」でした．回路設計の初体験にして，いきなり難度の高い設計が求められたことに，このとき気づきました．いちばんの問題は，細長い円柱形のロケットにどうやったら基板を押し込むことができるかでした．

普通の長方形の基板を作っても，ロケットに収まる気が

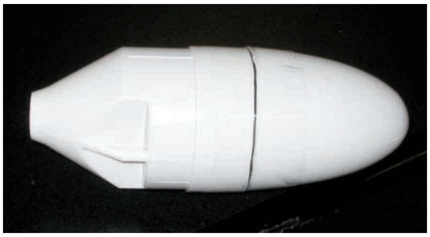


写真7　観測機器の搭載部分（ペイロード）

全長10cmにも満たない観測機器を搭載するロケットの先端部分．こんなものに3ヵ月半も振り回された．浜名湖で打ち上げることを前提としているため，防水性のことも考える必要があった．防水のために避妊具（コンドーム）を使用することを真剣に提案したが，みんなに冷たい目で見られた．結局，防水テープを使用することになった．

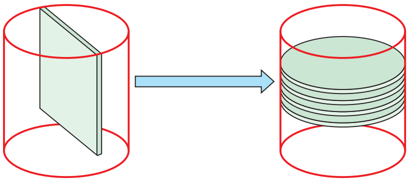


図2　ミルフィーユ基板構成

お菓子のミルフィーユみたいに何枚も重ねる構成ということで，ミルフィーユ基板と命名された．最初は冗談のつもりだったのだが，気がつくとその名まえで定着していた．

しません．散々悩んだあげくたどりついた答えは，「お菓子のミルフィーユみたいに基板を縦に積んでやればよいのでは？」ということでした（**図2**）．この構成ならロケットに収まりそうということで，ミルフィーユ基板の設計に取りかかりました．

3）基板サイズでロケット担当ともめる

基板を設計するために，まず製作する基板のサイズを決めなくてはなりません．基板のサイズはロケットの内径より1mm～2mm小さくする予定でいたのですが，ロケット担当から繰り返し「緩衝剤を入れるからもっと小さくしろよ！」と言われました．「ただでさえ小さい基板をさらに小さくさせる気か，コノヤロウ！」と，ロケット担当とは散々けんかをして，結局ロケットの内径から6mm余裕を持たせた直径3.4cmの円形基板を作ることになりました．

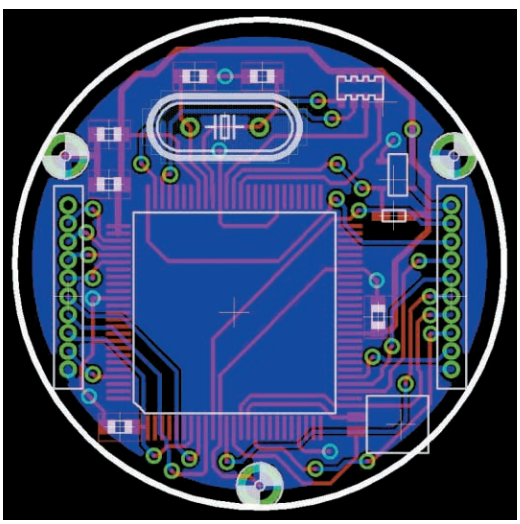


図3　作成したCAD設計図（基板レイアウト）

今回作成したのは両面基板で，表面が赤色のパターン，裏面が青色のパターンになっている．全体が青色で埋まっているのはベタ・グラウンドとしたため．

基板のサイズが決まったら次は部品の選定です．これには，最初思っていたよりもずいぶん苦労しました．というのは，小さい部品は高価なものが多いためです．上司の「高い！」ということばに頭を抱えながら，部品を選んでいきました．

4）"手はんだ"できないパターンを設計

回路を安定動作させるためにノイズ対策について勉強しました．ところが回路が小さすぎて，ノイズ対策を施すだけのスペースがまったくありません．パターンを引き回すだけで精いっぱいでした（**図3**）．

最初に設計したCAD図は，図面上では問題のないものだったのですが，手作業のはんだ付けが不可能な部分があり，却下となってしまいました．その後，何度も書き直し，やっとのことで完成させて，基板加工の工程に進みました（**写真8**）．

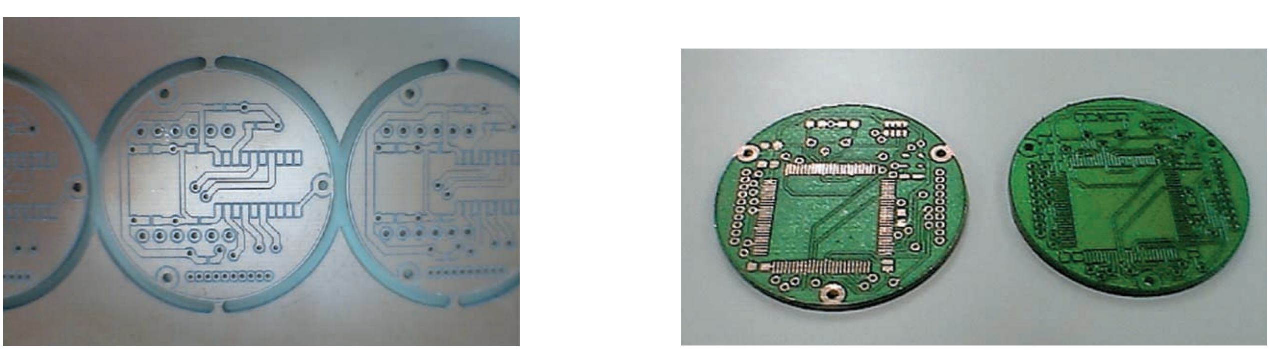
5）さあ，はんだ付けだ！

基板設計と基板加工が終わり，「さぁ，はんだ付けをしよう」というところで，ミスに気がつきました．地磁気センサ用基板に搭載されるICのピン・ピッチ（ピン間隔）が違っていたのです．結局，これは後に再設計ということになりました．また，実際にはんだ付けを行ってみると，はんだ付けが困難な部分が多く，最初に作成した基板の多くがはんだ不良によって動作しない，という事態になってしまいました．そのため，CAD設計からやり直すことになりました．

マイコンのピンのはんだ付けは，非常に苦労しました．はんだ付けが終わった時点で，購入したマイコンの半分以上が動作しない残骸となりました（**写真9**）．

6）これで完成？

予定より大幅に遅れましたが，なんとか回路ができ上が



（**a**）加工直後　（**b**）しあげ処理後

写真8　加工直後の基板としあげ処理後

今回はドリルを使用して銅板を削るタイプの基板加工機を使って加工した．加工後にレジストを使用し，それらしい基板に変身を遂げた．

りました．動作を確認したところ，ときどきプログラムが書き込めません．ノイズ対策をほとんど施せなかった影響がここで出ました．結局，正しく動作する基板は2組．なんとか動作する基板を作れたものの，課題が多く残る結果となりました（**写真10**）．

● デバイス・ドライバ担当：センサなどのドライバを作成

デバイス・ドライバ担当（後藤）の作業は，ロケットの加速度を測る加速度センサ，軸を補正する地磁気センサ，データを保存する記憶装置のEEPROM，保存したデータを読み出すRS-232-C（シリアル）の各ドライバを作ることでした．しかし，「デバイス・ドライバを作ったことがないので，何をしたらいいのかわからない」という状況で開発がスタートしたのです．

まずはセンサが動くかどうかを確認するためにデバッグ用ボードを使用して開発を進めることにしました（**写真11**）．

1）加速度センサのドライバの作成は簡単

今回使う加速度センサ（H48D，日立金属製）は，加速度に比例したアナログ電圧を出力します．そのため，制御用のマイコンにアナログ電圧を入力し，A-D変換を行ってディジタル信号を作ります．これは初心者でもできる，比較的易しい制御でした（**図4（a）**）．

2）地磁気センサとの通信に必要な割り込み制御は困難

今回使用する地磁気センサ（AM-45P，センセーション製）はA-D変換器を内蔵しています．そのため，マイコン側でA-D変換を行う必要はなく，シリアル通信でデータを出力します（**図4（b）**）．デバイスとの間でシリアル通信を行う際には最上位ビットから送信するか，最下位ビットから送信するかがデバイスによって異なります．そこでこれを確認する必要がありました．また，シリアル通信には送信割り込みと受信割り込みの制御が必要になります．これは初心者には難しく，制御することが困難でした．

3）EEPROMの書き込み方式と読み出し方式の違いで苦労

記録用のEEPROM（AT25HP512，米国Atmel社製）については，SPI（serial parallel interface）という通信規格の512kbpsに対応した品種を選びました．最初はI^2Cという規格に対応したものを選んでいたのですが，今回使用する制御用のH8マイコン（H8/3069F）ではI^2Cの通信ポート

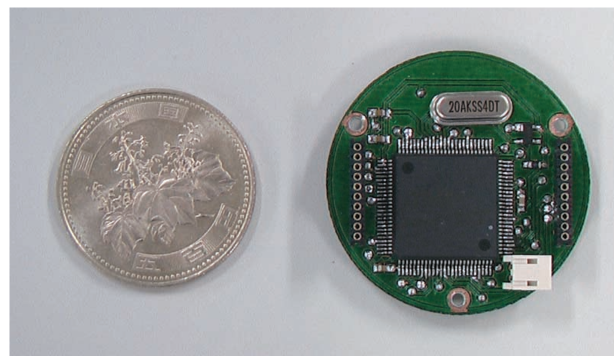


写真9　はんだ付けが完了した基板
正常動作した数少ない基板．いちばん初めにはんだ付けした回路は正常に動作しないうえ，あちこち焦げていた．その後，しだいにまともにはんだ付けができるようになっていった．左は大きさ比較用の500円玉である．

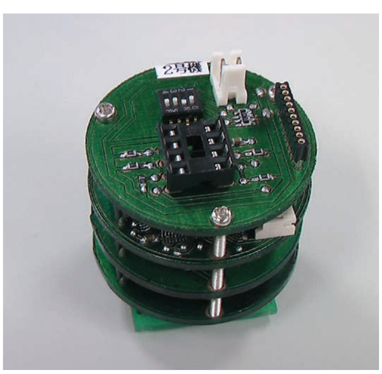
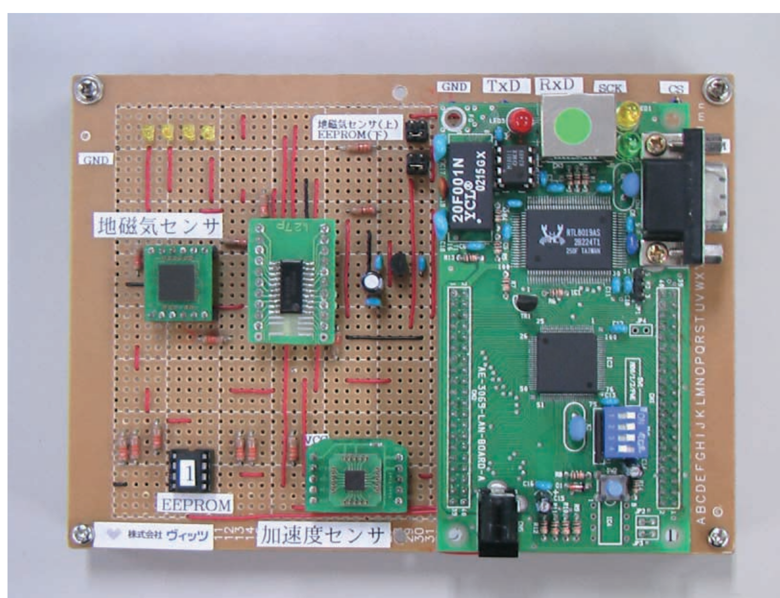


写真10　完成したミルフィーユ基板
結局，完成した基板は2組だけ．



▶ **写真11**
作成したデバッグ用ボード

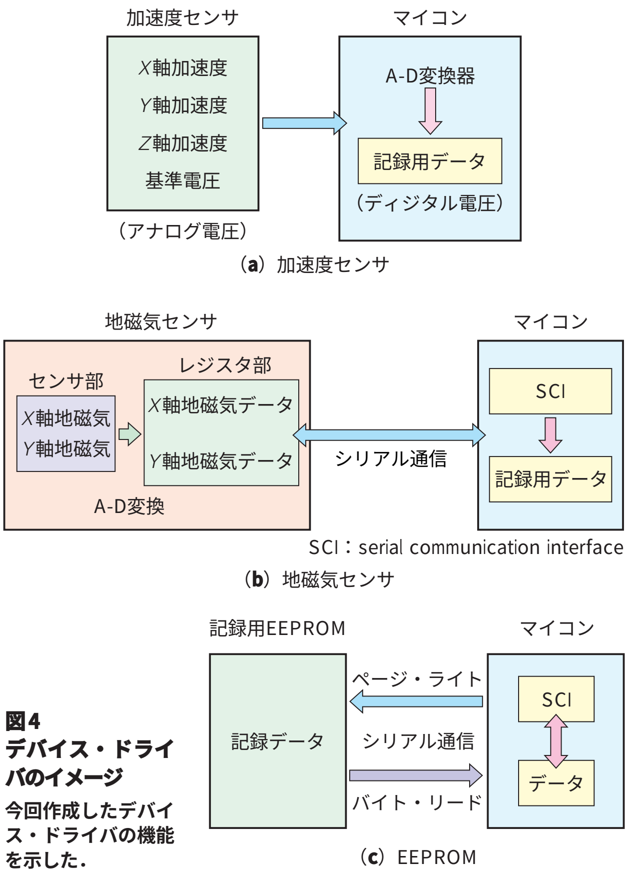


図4
デバイス・ドライバのイメージ
今回作成したデバイス・ドライバの機能を示した．

が使えません．もしI^2Cで通信しようとすると，汎用ポートで制御する必要があります．これは初心者には難しいという指摘を先輩エンジニアから受けました．

EEPROMに書き込む際には，128バイトの情報を一度に書き込むページ・ライトという方式を使用しています．そのため，センサから出力される値が128バイト分たまるまでは，マイコン上のバッファに一時的に保存し，128バイトたまったらEEPROMに書き込みを行います．逆にデータを読み出す際には，1バイトずつ情報を送るバイト・リードという方式でデータ転送を行っています(**図4(c)**)．

なぜ書き込みと読み出しで方法が違うのかというと，EEPROMがそのような仕様になっていたからです．最初，ページ・ライトができるのだからページ・リードもできるのだろうと思い込み，制御しようとしました．しかし思ったようにいかず，仕様書を読み返したところ，バイト・リードしかできないと書いてありました．この部分を修正するのに，かなり苦労しました．思い込みを排除して仕様書をよく読まなければならない，ということを身をもって思い知ったのです．



写真12　調査に使用した資料たち
「リアルタイムOSって何？」，「マイコンってどんなしくみ？」…．初心者レベルからのスタートだった．

● OS担当：地道に動作を確認しながらOSを移植

OS担当(沼田)の作業は，OSの移植でした．上述のようにセンサの制御にはH8マイコンを使用します．この上にTOPPERS/OSEKカーネルを搭載することになりました．

1) 資料と首っ引きでマニュアルを読む

今回移植を行うOSはすでに同じメーカのH8Sマイコンに対応していたため，「比較的簡単に移植できるんじゃない？」とのことでした．しかし，ミーティングでのこのことばをうのみにして，あとでたいへんな目に合うことになるとは思ってもいませんでした．

比較的簡単と言われても，まず何から手をつければよいのかまったくわかりません．先輩エンジニアからは「とりあえずH8Sマイコンについて調べてみて，次にH8マイコンとの違いをチェックして」というアドバイスをいただきました．そこでまず，H8Sマイコンについての調査を開始しました．が，H8Sマイコンのマニュアルを見たところ，その量はなんと1,300ページ．「全部読むのは無理じゃないか」と思いつつ，基本機能について読み進めていきました．しかし，マニュアルに書いてあることばがわかりません．マイコンは学生時代にほんの少し触れただけで，マニュアルを読むこと自体初めてでした．一つの説明文の中に専門的な用語を見つけてはその意味を調べ，大まかな機能を理解するという気の遠くなるような作業を2週間ほど続けていました(**写真12**)．マニュアルのマニュアルがほしい，と何度願ったかわかりません．

そのような作業を続けているうちに，「このままではとうてい移植なんて終わらない」という危機感を感じ，とりあえず既存のOSをH8Sマイコンの上で動かしてみることにしました．まず環境の構築です(**写真13**)．今回は開発環境として，Cygwin(Windows上でUNIXの機能を利用

できるフリーのソフトウェア開発環境）上でgcc（コンパイラ）を動かすことにしました．Cygwinをインストールした後，「さあ，やるぞ！」と初めてのmakeを行ったところ，「gccがありません」とのメッセージ．あれ，どうして？

きちんと手順どおりにgccも入れたのに，どうして「ない」って言われるんだろう．最初からこれではこの先どうなるのだろうと，かなりへこみました．しかし，落ち込んでいても作業は進みません．早速，先輩エンジニアにヘルプです．すると，「ちゃんとPATH通ってる？」とのおことば．PATHってなんですか？

PATHとは環境変数の一つで，コンピュータがプログラムを実行する際にそのプログラムの場所を記述したものを指します．つまり，プログラムの検索先です．このとき，「gccがありません」と言われてしまったのは，このPATHという環境変数にgccが置いてある場所を登録していなかったためでした．そのため，「gccの場所がわからない」とCygwinに注意されてしまったのです．先輩の説明を聞いてPATHを修正すると，今度はすんなりとコンパイルが完了．こんなことであっさり解決するなんて，とかなり拍子抜けでした．その後，H8Sマイコンにプログラムを転送し，実行しました．すると，プログラム内に記述した文字がパソコンの画面上に出現しました．動いてあたりまえなのですが，なぜか感動してしまいました．

2）1行1行，動作を確認

H8SマイコンでOSの動作を確認した後は，いよいよ移植作業に突入です．基本的な操作は，今回の移植のターゲットであるH8マイコンと，参考にするH8Sマイコンではほとんど変わりません．相違点は，マイコンそのものに依存する部分です．例えばレジスタです．機能は同じでも名まえが違ったり，H8SマイコンにはあるのにH8マイコンにはないレジスタが存在したりします．レジスタがない場合，プログラムで同じ機能を実行するように記述してカバーします．

マニュアルを調べつつ，ひととおりコードを書き上げた後は，デバッグ用ボードに転送して動作確認を行いました．ターミナル・ソフトの画面を開き，ファイルを転送し，後は実行コマンドを入力するだけです．緊張の一瞬です．画面をじっと見つめ，「GO」と実行用のコマンドを入力しました．ところが画面は真っ白．あれ？ 何にも出ない．あたりまえですが，移植作業はそんなに甘いものではありませんでした．

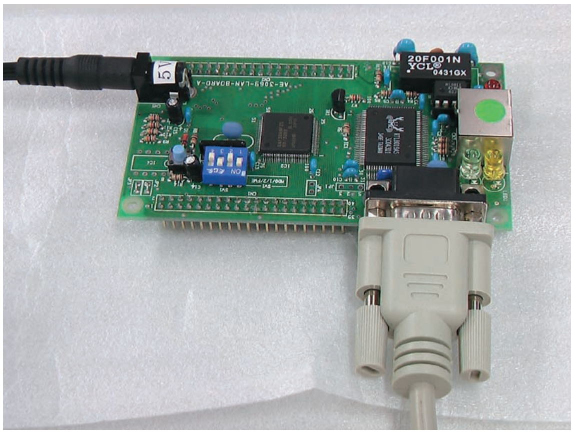


写真13　開発環境（H8/3069Fボード）
このようにシリアル・ケーブルでパソコンとデバッグ用ボードを接続し，移植作業を行った．

真っ白な画面に文字を出力させるため，地道でたいへんな作業が待っていました．まず，何が原因でプログラムが動作しないのかを調べることにしました．H8マイコンのポートに“H”と“L”の電圧を出力するデバッグ用のコードを書き，動作を確認したいコードの後ろに記述します．オシロスコープでポートの電圧を測定し，波形を確認できたら，また次のコードの後ろに記述．これをコード1行1行に対して行うという地道な作業を続けました．

今回の移植作業でいちばん悩まされたのは，割り込みでした．いくら設定しても，割り込みが入らなかったのです．先輩エンジニアにソースを見てもらったところ，「これ，ほんとに割り込み許可にしてる？」とあきれ顔．きちんと設定しているつもりだった担当者は，得意気に説明するのですが，説明の途中で「だから，違うって！」とさえぎられてしまいます．結局，わかっていなかった担当者に対して先輩エンジニアは何度も説明してくれました．理解力のない自分をほんとうに情けなく感じました．そんなふうに先輩に怒られ，「動かない！」と悩みながら，動作確認を進めていきました．

3）ついに移植成功

1行1行のコードの動作や割り込みの設定などを確認したあと，デバッグ用コードを取り除いて実行させてみました．すると，画面に1文字だけ文字が！ おお，これは完成が近いかもしれない．調べてみると，送受信用のバッファを壊してしまっていることが判明しました．早速直したところ無事動作するようになりました．やっと移植の完成です．サンプル・プログラムによりターミナル・ソフトに文

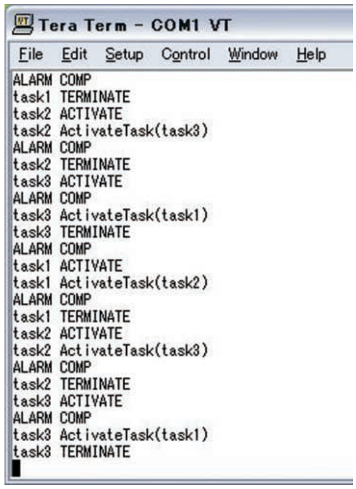


図5
サンプル・プログラムの実行画面
三つのタスクを順番に起動して終了する，というプログラム．「ACTIVATE」で起動，「TERMINATE」で終了を示している．「ALARM COMP」というのはアラームが満了したことを示す．このサンプルでは，周期的にアラームを満了させている．

字が出力された瞬間，思わず「動いた！」と声を上げてしまいました（**図5**）．

ここまでくるのに，ほんとうにいろいろなことにつまずきました．つまずかなかったことがないような気さえします．たった一つの機能がまちがっているだけで全体が動かなくなる，ということを肌で感じました．その1ヵ所にたどり着くまでがたいへんで，右往左往の連続でした．完成まではとても長い道のりでしたが，無事，OSを移植することができました．

● アプリケーション担当：基礎研修を軽視したつけが…

アプリケーション・ソフトウェア担当（京井）は，当初OS担当といっしょにH8SマイコンとH8マイコンの違いを調べていました．上記のOS担当のところでも述べられていますが，二つのマイコンのマニュアルを見比べても，ポイントがつかめていないので，何をやっているのかさっぱりわかりません．

とにかくわからない専門用語が多いと思いました．それも横文字やアルファベットの略語が多用されています．専門用語をインターネットなどで調べるのですが，これまたさっぱりです．このわからなさ過ぎる感覚は，英語を習い始めたころの感覚に似ていると思いました．日本語を読むのだから数学の式だけを見るよりも楽だろうなと思っていたのですが…．マニュアルを読む前は，よくあるビデオ・デッキの説明書とか，パソコンの説明書などをイメージしていて，「それくらいならプログラムを書けなくても読めるし，できる」と思っていました．実際に見てみるととんでもない．初めて見たとき，正直，「えっ！」と思いました．

まず，字が小さい．漢字が多い．それに加えてページ数の多さ．「500ページって何？ 多すぎない？ これを全部頭に入れなきゃならないの？」．

今考えると，実際にマイコンに載せるプログラムを触ってみて慣れてくると，マニュアルに書いてある内容はすごくよくわかります．しかし，この時点ではまだそれに気づいていませんでした．

思えば，4月の基礎研修（C言語やマイコンについての学習），そして5月のこのマイコンの調査という流れは，初心者にとっては比較的勉強しやすい構成になっていたと思います．C言語を学習して，実際に触ってみて，経験をもとに調査するという形になっていました．さらにその延長でリアルタイムOS（OSEK OSなど）についての講義があったのですが，4月の基礎研修を甘く見ていたせいもあり，担当者はもうお手上げでした．

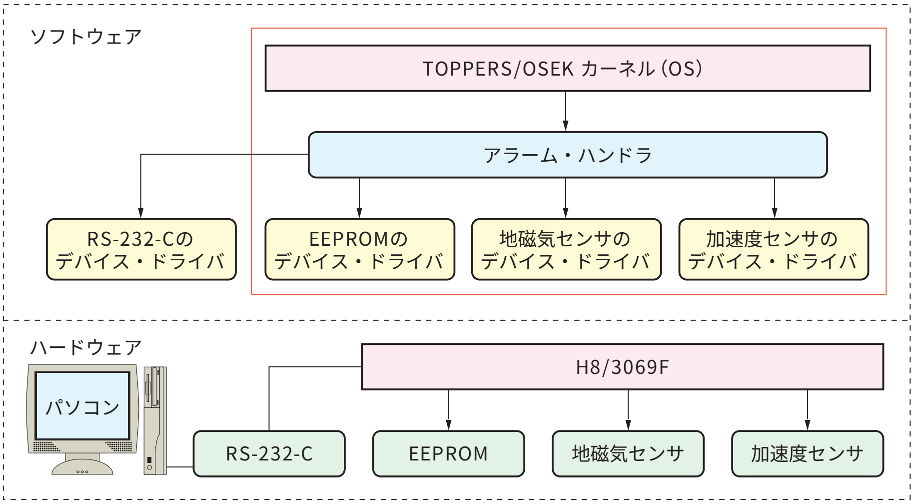
担当者があまりにもわかっていなかったためか，先輩エンジニアからは「アプリケーション・ソフトウェアの開発に専念して」という指示を受けました．「だいじょうぶ」，「そんなに難しくないから」と言われ，早速，調査に取りかかりました．

しかし，ここで問題が…．マニュアルやデータシートなどの資料がないのです．あるのはセンサとかEEPROMのマニュアルで，アプリケーション・ソフトウェア開発のマニュアルがありません．今ならユーザ・アプリケーションを作るためのマニュアルなんてものは存在しないとわかるのですが，当時は困惑しました．マニュアルを読むのは嫌だけど，「資料なしでどうやって作るんだろう？」と考えていました．

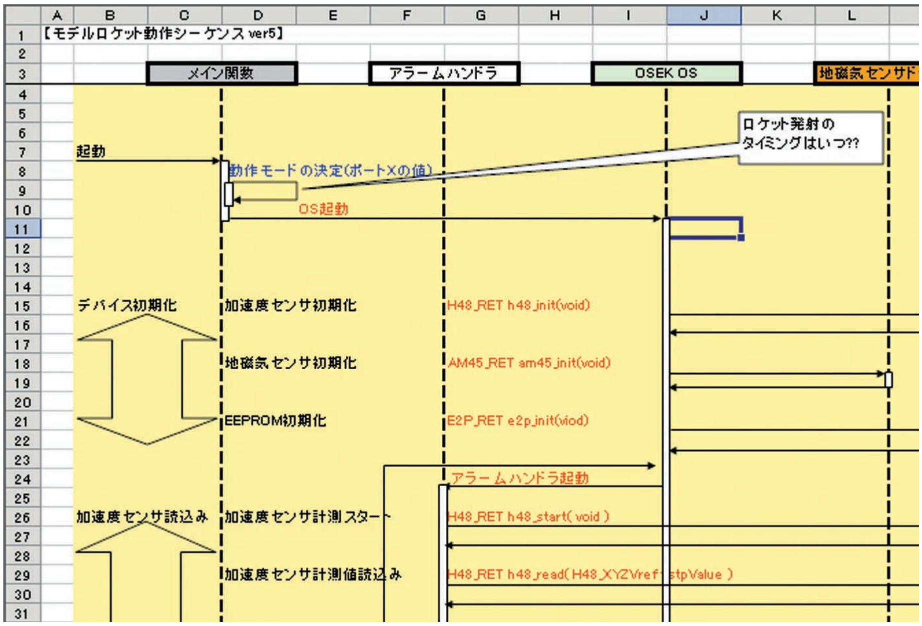
とりあえず，先輩エンジニアの指示に従い，第1ステップとして動作モジュール図というものを書きました（**図6(a)**）．どこがハードウェアの話でどこがソフトウェアの話かもわからないくらいでしたから，モジュールということばのイメージもなかなか理解できませんでした．ただし，ここまではだいたい先輩から教わったとおりだったので，まだそれほど時間はかかっていません．

次に動作シーケンス図を描きました（**図6(b)**）．ここでかなりの時間を要しました．真っさらな状態から書いたのではなく，先輩エンジニアに見本を書いてもらい，それをもとに作成しました．しかし，それでも何がなんだかさっぱり…．

ちなみにこのころ，担当者は入社前後の基礎研修の一部



（a）動作モジュール図



（b）動作シーケンス図

図6
動作モジュール図と動作シーケンス図
（a）は制御アプリケーションの動作モジュール図．やっと人前に出せるほどの形にしあがった．矢印の方向やモジュールの依存関係をわからないまま書いていた．（b）は制御アプリケーションの動作シーケンス図．

を怠けてしまって，基本的なプログラミングの知識が十分に身についていませんでした．そのため，プログラムの処理フローを追うことに慣れておらず，全体のイメージがないため，今から考えるとまったくちぐはぐなものになっていました．

3 Hamana-2.5 に突入

SWEST7の本番ではデータ計測ができないという非常に悔いの残る結果となりました．そこで帰社してから反省会を行った結果，自己啓発ということで，ある程度の結果が出るまでロケット・プロジェクトを続行するということになりました．この延長プロジェクトはHamana-2.5と呼ばれました．

Hamana-2.5では基板の安定化，デバイス・ドライバの完成，OSの開発環境の移植が課題となりました．ここからは自己啓発ということで，退勤後のお金にならない作業となりました．

● プリント基板を再設計

Hamana-2で使用した基板は動作が安定しなかったため，メンバで相談した結果，作り直しとなりました．しかし，

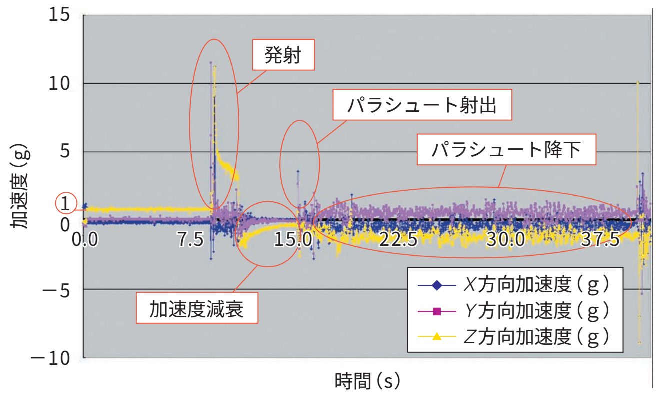


図7　Hamana-2.5 計測データ

ロケット打ち上げの際，最初に一瞬最大の加速度がかかり，そこからエンジンの噴射が弱まるにつれて緩やかに加速度が弱まっている．エンジンの噴射が終わると，空気抵抗による減速方向の加速度がかかっていることが読み取れる．その後，加速度が安定しないのは，パラシュートを展開したロケットが風に揺れながら落下しているためと考えられる．

マイコンなどの部品はすでに使い切っているものも多く，新たに購入しなければなりません．自己啓発の作業ということで，部品は自腹で購入することになります．ということで，新人全員でお金を出しあって部品を購入しました．

部品を買い，CAD 図を書き直してはみたものの，通常業務が忙しく，なかなか基板加工を行えません．結局，Hamana-2.5 用の基板が完成したのは，打ち上げ予定日の数日前となってしまいました．

● 開発環境を Pizzafactory と統合

Hamana-2.5 では，OS を TOPPERS/JSP カーネルの開発環境である Pizzafactory（もなみソフトウェア製）に対応させることにしました．Pizzafactory は GNU 開発環境であり，gcc と TOPPERS/JSP カーネルが組み込まれています．また，ユーザが操作できる階層がカーネルから独立しているので，インストールした後，すぐにサンプル・プログラムを実行できます．これで環境構築の手間が減ります．

Hamana-2 プロジェクトでは，加速度センサのデバイス・ドライバの開発に Pizzafactory を使用していたのですが，Cygwin で開発した OSEK カーネルと結合した際に，開発環境の違いにより文字コードなどのエラーが発生していました．その修正は，時間の浪費としか思えないくらい泥くさい作業だったので，絶対に開発環境を統合するべきだと考えました．

Cygwin と Pizzafactory の違いは PATH が違っているという点です．Pizzafactory はユーザが操作できる階層がカーネルから独立しているので，ユーザ階層からカーネル，またはその逆への PATH を設定するのがたいへんでした．数十ヵ所あった PATH をすべて手書きで書き換えることにより，OSEK を Pizzafactory 対応にできました．

● デバイス・ドライバが完成

Hamana-2 では正常に書き込めなかった EEPROM のデバイス・ドライバの完成を第 1 とし，できれば地磁気センサのデバイス・ドライバも完成させたいと考え，デバイス・ドライバの見直し作業を始めました．

ところが，SWEST 本番前にはあれだけ動作しなかった EEPROM が，落ち着いてプログラムを見直し，修正を行ったところ，無事動作しました．ただし，地磁気センサのデバイス・ドライバの完成までは手が回らず，Hamana-2.5 では加速度センサによる記録のみとなりました．

● Hamana-2.5，ついに打ち上げ

SWEST7 が終了してから 2 ヵ月たった 10 月．ようやく Hamana-2.5 の打ち上げを行うことができました．久しぶりのロケット打ち上げということで，念入りに準備したはずだったのですが，ロケット打ち上げに必要なランチロッド（打ち上げの際にロケットの打ち上げ方向を誘導するための棒）を忘れてきてしまいました．何か代わりになるものはないかということで，急きょ衣服をかけるハンガを加工してランチロッドにするという荒わざで何とか打ち上げを行いました．

その結果が図 7 のグラフです．最後までアクシデントが尽きませんでしたが，これにて Hamana-2.5 プロジェクトは終了となりました．この経験を生かし，今年開催予定の Hamana-3 では，同じような失敗をしないように今年の新人をサポートしていきたいと思います．

わたなべ・ともひろ，すぎやま・あゆむ，ぬまた・あみ，
ごとう・こういち，きょうい・まさゆき
（株）ヴィッツ　開発第 3 部